23rd AIAA Computational Fluid Dynamics Conference

Lattice Boltzmann and Navier-Stokes Cartesian CFD Approaches for Airframe Noise Predictions

Michael F. Barad*[1], Joseph G. Kocheemoolayil*[2], and Cetin C. Kiris*[1]

[1]*NASA Ames Research Center, Moffett Field, CA 94035*
[2]*Science and Technology Corporation, Moffett Field, CA 94035*

Lattice Boltzmann (LB) and compressible Navier-Stokes (NS) equations based computational fluid dynamics (CFD) approaches are compared for simulating airframe noise. Both LB and NS CFD approaches are implemented within the Launch Ascent and Vehicle Aerodynamics (LAVA) framework. Both schemes utilize the same underlying Cartesian structured mesh paradigm with provision for local adaptive grid refinement and sub-cycling in time. We choose a prototypical massively separated, wake-dominated flow ideally suited for Cartesian-grid based approaches in this study - The partially-dressed, cavity-closed nose landing gear (PDCC-NLG) noise problem from AIAA's Benchmark problems for Airframe Noise Computations (BANC) series of workshops. The relative accuracy and computational efficiency of the two approaches are systematically compared. Detailed comments are made on the potential held by LB to significantly reduce time-to-solution for a desired level of accuracy within the context of modeling airframe noise from first principles.

I. Introduction

Airframe noise is the noise that is generated by non-propulsive components of an aircraft. It can be divided into noise sources from: wings, including tail surfaces and fuselage; high lift devices, including leading edge slats, flap side edges, and brackets; and undercarriage, which includes wheels, axles, legs/struts, fairings, brake cables, pipes, wheel wells, and doors.[1] This noise is a nuisance in the vicinity of both major and minor airports throughout the world, and is a major focus of manufacturers designing, retrofitting, and operating current and future aircraft. There is increasing evidence that airframe and engine noise are comparable in the approach to landing phase. Thus efforts to reduce aircraft noise further should necessarily target ways to reduce airframe noise in addition to jet noise, turbofan noise and core noise.

Computational prediction of airframe noise is based on lower-fidelity empirical approaches, or increasingly, based on computational fluid dynamics (CFD) which is often coupled with an acoustic analogy. The focus of this work is on high-fidelity CFD approaches, and specifically on accurate methods with a fast turn-around time. More complete overviews of airframe noise prediction approaches is available in the literature.[2,3]

CFD has achieved global acceptance as a mature discipline that complements traditional wind tunnel and flight testing. In a modern, fast-paced design environment where decisions are tightly supported/driven by extensive simulation data, increasing pressure is being exerted on CFD practitioners to improve geometric fidelity and reduce turnaround times. This trend represents a paradigm shift that favors efficient and versatile CFD frameworks in place of specialized legacy CFD solvers typically optimized for a limited set of applications. The emphasis on CFD simulation turnaround time highlights several bottlenecks in the simulation process, most significant of which is the preparation of the computational geometry and its volumetric meshing.

Several different meshing and numerical discretization strategies such as structured Cartesian[4–11] and unstructured[12,13] have emerged as alternatives to the classical structured curvilinear[14,15] approach (see Kiris et al[16] for examples). Structured curvilinear grid generation is extremely time-consuming and labor-

*Computational Aerosciences Branch, NAS Division, MS N258-2, AIAA Senior Member

intensive, usually requiring significant user expertise, particularly for complex geometries of engineering importance. This motivates approaches that require little investment in grid generation.

Structured Cartesian methods are essentially free of labor-intensive manual volume mesh generation, are computationally efficient, and well-suited to implementation of higher-order methods and Structured Adaptive Mesh Refinement (SAMR).[4,5,17] However, resolving boundary layers at full-scale Reynolds numbers remains prohibitively expensive due to the isotropic nature of the mesh. Cartesian methods (block structured and otherwise) are particularly well-suited for inviscid design optimization applications and have attained widespread adoption.[10,11] On the other hand, viscous simulations at full-scale Reynolds numbers relevant to the aerospace industry using Cartesian methods remain challenging.

While NS based simulations have been the bread and butter of CFD practitioners since the birth of the field, LB based approaches have been gaining traction recently due to their attractive properties.[18–21] The benefits of LB methods (LBM) include ultra high performance,[22] minimal numerical dissipation, and the ability to simulate arbitrarily complex geometry in a relatively straight-forward manner. High performance is due to significantly lower floating point operations relative to high-order Navier-Stokes algorithms, excellent data locality, ability to vectorize and multithread the underlying operations on current and near-future hardware (despite the memory-bound nature of the algorithm), and excellent, more or less platform-agnostic parallel scalability. Minimal numerical dissipation is critical for computational aeroacoustics and ideal for large eddy simulations (LES). It is well-known that turbulence is much more sensitive to dissipative errors than dispersion errors.

In this work we aim to systematically quantify the advantages and disadvantages of our LB and NS schemes as implemented within a single software framework (LAVA). To avoid distractions related to inherent limitations of the paradigm itself, we focus on undercarriage-generated airframe noise which is an application that is ideally suited to Cartesian-grid methods. In this paper, we focus on accurate prediction of surface pressure fluctuations at critical locations on the landing gear, which are ultimately responsible for far-field noise. Far-field noise predictions will be reported elsewhere.

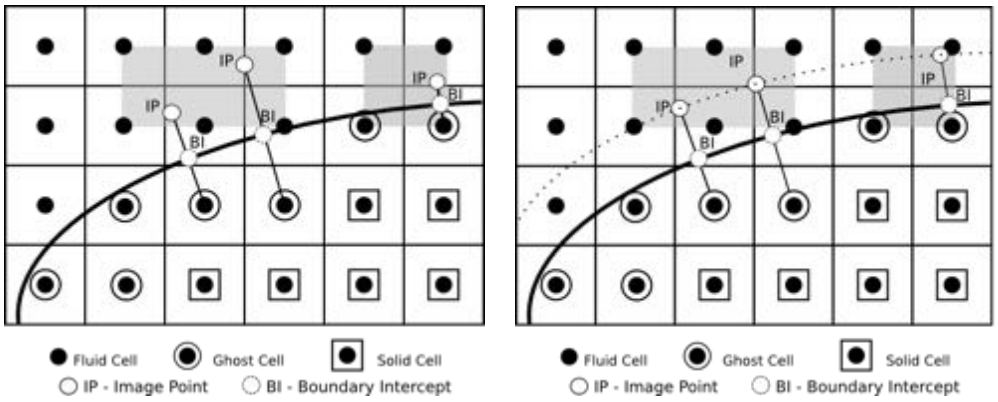
II. Methods

II.A. Navier-Stokes Solver

For the compressible Navier-Stokes equations, two sharp immersed boundary methods are available in LAVA Cartesian. The Cartesian immersed representation makes it possible to easily model complex geometry without having to manually generate a volume grid. The first immersed boundary method is based on a ghost cell boundary treatment,[23,24] referred to as ghost cell method, or GCM, and the second utilizes only interior (i.e. in the fluid) points with locally stabilized higher-order finite-difference stencils,[25,26] referred to as immersed interface method, or IIM. Both schemes have been extensively documented.[16] In the following discussion both methods are very briefly described.

The ghost cell method is based on filling cells that lie inside the geometry with appropriate flow data. Figures 1a and 1b schematically display the procedure for filling the ghost cells. For a second-order accurate scheme, only the first layer of grid points beyond the boundary of the geometry needs to be filled and are referred to as ghost cells, while the rest of the grid points inside the geometry are referred to as solid points. IB ghost cells are covered cells that are within the stencil of a fluid cell, and are filled using Image-Points (IP) and boundary conditions. Two variants for defining IP locations are available: exact reflection (Figure 1a) and fixed distance, typically $|\Delta\vec{x}|/2$ (Figure 1b). A higher-order extension of this method by enlarging interpolation and difference stencils is theoretically straightforward. In practice, however, the robustness of the numerical scheme suffers severely when this extension is applied.

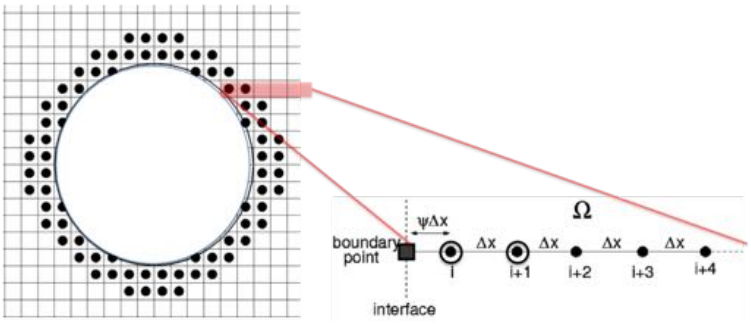
While GCM is relatively easy to implement, the IIM is higher-order accurate and does not require ghost cells inside the geometry, critical for thin geometries. It is possible to implement GCM for thin geometry by utilizing complex data-structures, but only naive ghost cell based GCMs (as indicated in Figure 1) are considered here. In the higher-order method, irregular one-sided finite-difference stencils carry a free parameter that is adjusted locally to improve the stability characteristics of the finite difference scheme. The irregular finite difference stencils are treated in isolation from the rest of the computational domain as shown in Figure 2b. The validity of isolating the local grid stencils has previously been demonstrated.[25,26] Figure 2a shows the irregular finite difference stencil in the vicinity of the immersed boundary with a normalized distance to the grid line intersection point of ψ. The boundary conditions are provided at grid line intersection points, and they are obtained by weighted least-squares interpolation/extrapolation stencils. For slip wall



(a) (b)

Figure 1. GCM schematic: ghost cell filling using two different IB methods: (a) full image reflection,[23] (b) fixed distance reflection into fluid (indicated by dotted line). Image-Points (IP) are interpolated. IP data is combined with boundary condition and the resulting ghost cell values are computed. With (a) IP can get arbitrarily close to the ghost cell, and the interpolation stencil for IP can be dominated by the ghost cell. With (b) the fixed stand-off distance forces the interpolation stencil to have significant contributions from fluid cells.

boundary conditions, the velocity field is also extrapolated towards the grid line intersection points while the no-penetration condition ($\boldsymbol{v} \cdot \boldsymbol{n} = 0$) is imposed at the solid wall.



(a) (b)

Figure 2. IIM schematic: (a) Localization illustrated in 2D for circle immersed into Cartesian grid. (b) Computational setup in the vicinity of the immersed boundary including irregular grid points (⊙), points in fluid (●), boundary points (■). Points outside the computational domain are dropped.

One of the key challenges for immersed boundary methods is the treatment of viscous walls at high Reynolds numbers. For high-Reynolds number viscous flow simulations, current immersed boundary approaches are inefficient in resolving the viscous boundary layers since the Cartesian mesh layout generally does not allow the use of (wall-normal) high aspect ratio cells in the vicinity of the wall. This issue has been addressed in several research investigations mainly by employing wall models near the immersed boundary.[27] An equilibrium wall-stress model is now available in LAVA Cartesian and can be used to account for the unresolved boundary layer and turbulent eddies in the vicinity of the wall.[28–31] All simulations in this paper are run with no-slip conditions, validation using the wall-stress model is left for future work.

II.B. Lattice Boltzmann Solver

Over the past two decades, the Lattice Boltzmann Method has evolved into a mature technique for simulating engineering fluid flows of practical importance. The LB method is a mesoscopic approach wherein simplified kinetic equations that retain just enough detail to satisfy the desired macroscopic equations of fluid motion (weakly compressible, isothermal Navier-Stokes equations in the present context) are solved.[18] The local state of fluid motion is described by density distribution functions $f(\vec{x}, t, v)$, which upon being normalized by the local density represent the probability of finding particles moving with velocity v in an infinitesimal volume dx about $\vec{x}$. The familiar macroscopic variables such as density and the components of momentum are determined from the density distribution functions through moment summations.

II.B.1. Governing Equations

The LB equation governs the space-time evolution of density distribution functions $f(\vec{x}, t, v)$. The density distribution function $f(\vec{x}, t, v)$ at a particular node of the lattice defines the fraction of mass contained in a control volume surrounding the node which moves at velocity v. For example, $f(\vec{x}, t, 0)$ defines the fraction of mass contained in the control volume which is at rest. The equation is solved numerically through an extremely efficient *collide at nodes and stream along links* algorithm. In a single time step the virtual, computational particles collide at a node relaxing towards the local *equilibrium* and subsequently hop on to the neighboring nodes of the lattice. The velocity space is commonly discretized into 15, 19 or 27 speeds on a regular cubic lattice in 3D, where the notation $DdQq$ describes a lattice in d dimensions with q discrete velocities. The local equilibrium is a truncated, low-speed approximation of the Maxwell-Boltzmann distribution function corresponding to the local macroscopic variables.

The collision operation is perfectly local in space and time. Every node performs this operation independent of every other node in the lattice. It is perfectly non-local in the velocity space, i.e. all particles talk to each other during the collision process at the node. The velocity space is discrete. The particles are only allowed to have a certain number of discrete velocities consistent with discrete node-to-node streaming operations on the lattice. The particles are bound to the lattice. They are not allowed to occupy positions in between the nodes of the lattice. They simply hop from one node to another on the lattice as time advances. This makes exact, dissipation-free advection (i.e. streaming) a simple copy operation.

The fluid viscosity sets the rate at which the density distribution functions relax to the local equilibrium. More viscosity implies faster relaxation to the local equilibrium and lower deviations from the local equilibrium on average. The relaxation rate is also inversely related to the local lattice spacing. A coarser mesh results in larger deviations from the local equilibrium for a given viscosity. For small Mach numbers, the Chapman-Enskog expansion can be used to show that the simplified collision rules with truncated equilibrium distribution functions reproduce the weakly-compressible, isothermal Navier-Stokes equations in the low-frequency/large-wavelength limit.[32]

II.B.2. Collision Models

The following collision models are currently available in the solver:

1. Bhatnagar-Gross-Krook (BGK) model - All distribution functions relax to their local equilibrium values at the same constant rate for every node in a given grid refinement level.[33] All moments of the distribution function (from order two to six) relax to their equilibrium values at a single rate that is only a function of the fluid viscosity and the local lattice spacing.

2. Bhatnagar-Gross-Krook model with realizability constraint - The rate at which the distribution functions relax to their local equilibrium is modified *locally* at every node of the lattice to enforce strict realizability of the distribution functions everywhere.[34] Strict enforcement of realizability through local under-relaxation (if and where necessary) is tantamount to surgical addition of artificial viscosity in regions where the simulated flow is too under-resolved to maintain numerical stability. This approach guarantees nonlinear stability and makes engineering Reynolds numbers accessible at a manageable computational cost.

3. Entropic model - The rate at which the distribution functions relax to their local equilibrium is modified *locally* to enforce compliance with the H-theorem.[35] The local relaxation rate is modified to ensure that the Boltzmann-Shannon entropy of the post collision state is not lower than the pre-collision state.[35] Since the Boltzmann-Shannon entropy is a Lyapunov function in the phase space corresponding to the density distribution functions, compliance with H-theorem guarantees nonlinear stability. Compliance with H-theorem also guarantees realizability by construction.

4. Multiple-Relaxation Time (MRT) model - The collision operation is performed in the moment space. Conceptually, the pre-collision density distribution functions are transformed into the moment space through a linear transformation and post-collision moments transformed back into the velocity space. The higher-order moments (the so-called *ghost* modes) are relaxed to equilibrium at a faster rate relative to the hydrodynamic modes.[36]

5. Entropic Multiple-Relaxation Time (EMRT) model - Instead of choosing the relaxation rates for the higher-order moments in an *ad hoc* manner for the entire lattice, they are chosen locally on a node-to-node basis based on the entropic principle.[37] All the LB simulations presented in this paper were performed using the entropic multi-relaxation time model on the D3Q27 lattice.

6. Regularized Bhatnagar-Gross-Krook model - The pre-collision state is *regularized* by equilibrating

the higher-order moments using Grad's approximation.[38] The post-collision state is obtained from the regularized pre-collision state using the standard Bhatnagar-Gross-Krook (BGK) rule.

The unresolved, subgrid scales of turbulence are modeled using the constant-coefficient Smagorinsky model,[39] which modifies the local rate at which the distribution functions relax to equilibrium.

II.B.3. Boundary Conditions

The LBM formulation accounts for the presence of an embedded geometry through simple, intuitive rules which express the unknown, *incoming* populations in terms of the known, *outgoing* populations in order to complete the streaming step for lattice links intercepted by the geometry. For example, simply bouncing the particles back from the geometry in the opposite direction enforces the no-slip boundary condition. Specular reflection of particles in the mirror image direction is realized as the free-slip/no-penetration boundary condition. Both of these standard bounce-back (SBB) rules approximate the actual curved geometry with a series of small steps whose sizes are proportional to the local grid spacing.

Interpolated bounce-back rules capture the curvature in geometry more accurately, albeit at the cost of forfeiting strict realizability and precise mass conservation. Our experiments indicate that the linear bounce-back (LBB) formulation proposed by Bouzidi et al.[40] is accurate, sufficiently robust, and extremely efficient computationally. The LBB computational time penalty relative to the SBB rule is minimal for the Structure-of-Arrays (SoA) data structure we have employed in our solver, where the boundary condition routine can take advantage of excellent cache reuse.

A numerical sponge, where the relaxation time is ramped up, was used in a buffer zone close to far-field boundary to minimize spurious reflections.

II.C. Multi-Resolution Cartesian Methods

The Cartesian SAMR methodology is capable of automatically generating, refining, and coarsening nested Cartesian volumes given a closed surface triangulation, and hence, offers the ability to dynamically track important flow features as they develop. Figure 3a shows an idealized SAMR hierarchy refining about a feature. In adaptive methods, one adjusts the computational effort locally to target a uniform level of accuracy throughout the problem domain. Cartesian SAMR is a proven methodology for multi-scale problems, with an extensive existing mathematical and software knowledge base.[5–9,41] The LAVA code incorporates data structures and inter-level SAMR operators from the high-performance Chombo library.[42] SAMR allows the simulation of a wide range of spatial and temporal scales through local refinement.

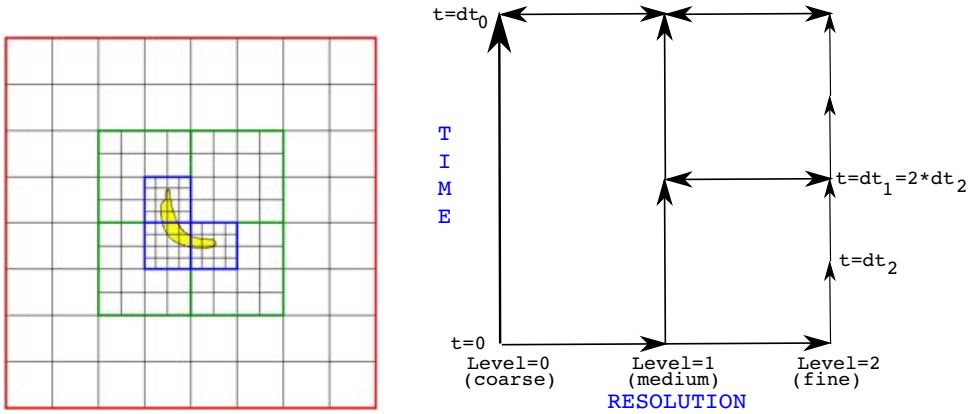
II.C.1. Time-Stepping

The advancement of time with a locally refined Cartesian mesh hierarchy is typically either composite or sub-cycled. For the composite approach, a constraint is imposed that all SAMR levels are advanced with the same time step, i.e. Δt = constant. The composite approach is not possible for the LB formulations considered here. For the sub-cycling of levels approach, as is done in the literature,[6,43] each level is advanced in such a way as to maintain a constant CFL, i.e. $\Delta t = C\Delta x$.

Figure 3b illustrates the flow of information during the subcycling algorithm. The right pointing arrows indicate space-time interpolation from a coarse level to a fine level, filling ghost cells. The left arrows indicate an averaging down procedure from fine to coarse levels. The up arrows indicate a single level advancement in time. The inter-level interpolation and restriction operators are second-order accurate.

II.C.2. Complex Geometry

Some of the key challenges for simulating extremely complex geometries, such as the landing gear, is the treatment of thin geometry, thin boundary layers near solid walls, and relative body motion. Several fast algorithms for discrete geometry manipulation are required for an efficient Cartesian solver. Specifically, geometric queries from the closed surface representation, including inside-outside, and signed distance (exact and approximate) functions. For simple analytic geometries, a Constructive Solid Geometry (CSG) methodology[44] is exact and efficient. For complex geometries where CSG is not available, geometries are represented with closed surface triangulations. In what follows, closed surface triangulations representing the surface are assumed.



(a) (b)

Figure 3. (a) Block structured adaptive mesh refinement showing 3 levels refined by factors of 2. Red is level 0 (coarse), green is level 1 (medium), and blue is level 2 (fine). (b) Recursive sub-cycling algorithm for a 3 level hierarchy of grids. Arrows indicate direction of information propagation.

The treatment of complex geometries with Cartesian methods can be viewed as a discretization of the geometry onto a multi-resolution Cartesian mesh. For both LB and NS Cartesian approaches this can be distilled down to computing stencil clouds in the vicinity of the geometry for special treatment by the appropriate operators. These stencil clouds are composed of valid points usable by the discretization that are guaranteed to maintain certain properties (e.g. water-tight) required for simulation. The algorithms ensure that points are only selected for the point cloud if they reside on the same side of the boundary as the irregular grid point. Figure 4a schematically illustrates the cloud selection process at an irregular grid point. First, the base clouds for all irregular grid points are computed while the base clouds for the regular grid points are not computed as they are known by definition to contain the direct neighbors, i.e., $(i\pm1,j,k)$, $(i,j\pm1,k)$, and $(i,j,k\pm1)$. Then the point cloud at each irregular grid point is gradually extended by adding the base clouds from its valid neighbors. The moving geometry case is not considered in this work, but has been addressed by the authors.[45]

As the LB and NS based Cartesian schemes depend closely on the geometry description, high-performance high-accuracy geometry queries are required for quick turnaround, reliable simulations, especially when the geometry is moving or adaptive meshing is used. The required high-performance geometry queries necessary for our immersed Cartesian discretizations include: 1) point inside/outside, 2) ray-surface intersections, 3) nearest point, and 4) box-surface intersection tests. Several approaches to serving these geometry kernels are possible. Two approaches were considered here: 1) level-set based, and 2) triangulation based. Level-set based kernels are dependent on accurate distances to the geometry, either based on analytic functions or by initialization using a triangulation based kernel.[44,46,47] While level-set based kernels have elegant mathematical and performance properties, they suffer from both complexity of interpolation algorithms in the vicinity of thin geometries and degradation of accuracy for moving body problems due to implicit mollification. For these reasons, we focus on exact triangulation based geometry kernels for the remainder of this work. For triangulation based geometry kernels, there are currently two options to choose from in the solver: 1) in-house hand-coded routines, and 2) hardware vendor supplied highly optimized routines. Our initial efforts relied on an in-house implementation of the required geometry kernels, but the performance was not sufficient for large scale cases, especially those with moving geometry where $\mathcal{O}\left(M * N^{2/3}\right)$ queries are made for a computation with N cells and M timesteps.

For this work we rely on the highly optimized Embree ray-tracing library.[48] We utilize the Embree data-structures, which are highly vectorized and utilize OpenMP threading. Embree's internal vectorized bounding volume hierarchies (BVH) data-structures are designed for highly-optimized ray traversal and triangle intersection tests. The box based BVH tree data-structure is chosen within Embree due to the small memory footprint, quick traversals, and minimal setup cost. Within the BVH, the branching factor is fixed at four (i.e. each node has four child nodes) to enable SIMD vectorization. For each single-ray

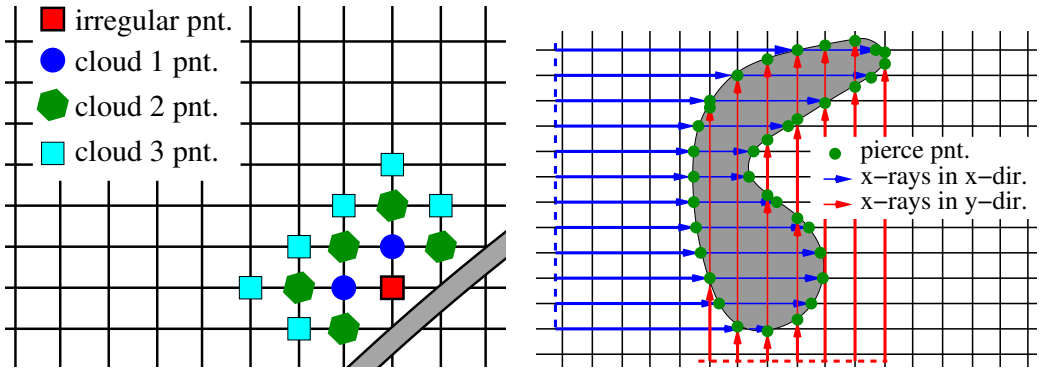


Figure 4. (a) Cloud selection process for the boundary extrapolation stencil, (b) illustration of x-ray algorithm used for in-out testing.

query, Embree is able to optimally traverse the BVH with vectorized instructions, where the ray is tested for intersection against the four child nodes in parallel, drilling down to the BVH leaf nodes. Ray intersections against multiple triangles can be compared in parallel. Separate channels (e.g., x, y, z) of a single ray-triangle intersection test can also be computed in parallel. Multiple rays can be queried as a packet within Embree, exploiting locality, but we have not yet evaluated this feature.

We interface our solver directly with the query functions of the four kernels mentioned above, which are all based on Embree's BVH. 1) Point inside/outside queries are computed by ray-tracing from the query point to the far field and counting crossings. An odd count indicates a point inside the geometry, and an even count indicates a point outside the geometry. 2) Ray-surface intersection queries are handled directly in Embree. For example, Figure 4b illustrates the ray-surface intersection queries that are used to determine the grid intersections and the resulting surface normals at the intersection pierce points. 3) Nearest point queries are evaluated by traversing the BVH directly and evaluating box distances first and subsequently leaf node point-triangle distances. 4) Box-surface intersection tests are also evaluated as a BVH traversal, where box-box tests are computed first, and box-triangle intersection tests are subsequently computed on the leaf node triangles.

III. Results

In what follows we describe simulations of flow past a landing gear. Specifically, we are interested in the partially-dressed, cavity-closed nose landing gear (PDCC-NLG) noise problem from AIAA's Benchmark problems for Airframe Noise Computations (BANC) series of workshops.

III.A. Setup

Conditions and geometry were chosen to match experiments conducted in the open-jet University of Florida Aeroacoustic Flow Facility (UFAFF). No attempt was made to include the open-jet facility effects, including the incoming jet and the associated shear layers and collector. The acoustic absorbing effect of the surrounding anechoic chamber, while not explicitly modeled in the simulation, is mimicked through the action of non-reflecting boundary conditions. The simulation assumes free-stream conditions as recommended by the workshop committee. The mounting wall is also not modeled in the simulation beyond the leading edge of the fuselage.

Table 1 lists the key cases run by the LAVA Cartesian codes for this work. Both Navier-Stokes (NS) and Lattice Boltzmann (LB) based approaches were utilized for the simulations. In the table, the RunID indicates NS or LB and the number indicates the wake resolution used. The NS simulations were run before the LB development effort within LAVA began (based on the GCM approach) and utilized a slightly different mesh than the current LB simulations. We currently have a constraint that successive refinement levels can only differ by a factor of two in addition to the proper nesting requirements in our LB implementation (but not in the NS implementation). This precluded a precise one-to-one comparison on *identical* meshes. The NS simulation utilized finer cells in the immediate vicinity of the geometry than the LB simulations, leading

Method / RunID	CPU Cores	Cells ($\times 10^6$)	Surface Δx[m]	Wake Δx[m]	Surface IO Freq.	Volume IO Freq.	Wall Days	Core Days	Relative SBU Expense
NS-3-GCM	3000*	298	2.44e-4	9.76e-4	10	50	20.5	61352	**12.1**
NS-3-IIM	9600†	222	9.76e-4	9.76e-4	10	50	6.1	58490	**15.3**
LB-1	700	55.9	3.91e-3	3.91e-3	1000	50	0.16	112	0.035
LB-2	1400	90.9	1.95e-3	1.95e-3	1000	50	0.41	574	0.182
LB-3	1400	260	9.76e-4	9.76e-4	1000	50	1.5	2100	0.665
LB-3-IO	1400	260	9.76e-4	9.76e-4	10	50	2.25	3156	**1**
LB-4	2800	1603	4.88e-4	4.88e-4	1000	50	8.9	24920	7.90

Table 1. Grid and Performance Statistics: comparing Navier-Stokes (NS) and Lattice Boltzmann (LB) based simulations. All runs utilized Broadwell cores on NASA's Pleiades supercomputer, except as noted $\star$ where Ivy Bridge cores, or $\dagger$ where Haswell cores were used. Relative expense takes into account processor performance ratios: Broadwell vs Ivy Bridge where a factor of 4.04/2.51 = 1.61 is applied, or Broadwell vs Haswell where a factor of 4.04/3.34 = 1.21 is applied. These ratios are as per standard billing units (SBU) on Pleiades. Run durations go to 0.19 physical seconds.

to a slight discrepancy in total cell counts for comparison runs of similar resolution (298M vs 260M for NS-4 vs LB-3-IO in Table 1). Surface and volume output frequencies (measured in timesteps) are also indicated in Table 1.

Computational meshes used in the simulations are shown in Figure 5.

III.B. Near Field Power Spectral Density

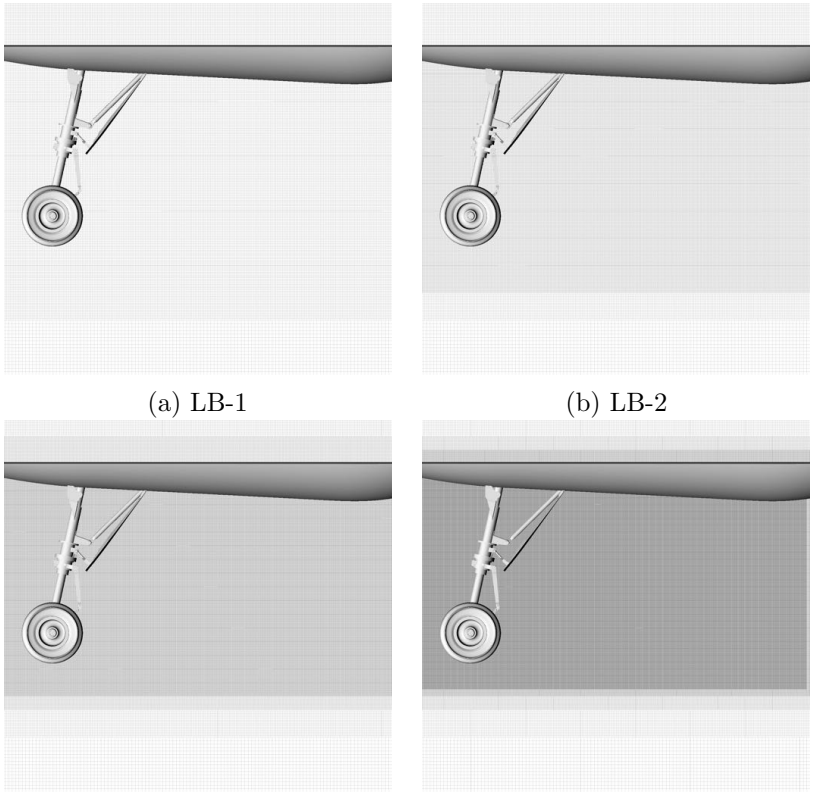
Unsteady pressure observations were taken at several locations in the experiment, as shown in Figure 6. At those locations, spectral analyses of both experiment and CFD results were logged. Figure 7 presents power spectral density (PSD) of sound at different locations, and the PSD is compared to experimental data and to CFD submissions at the BANC-III workshop.[49] PSD analyses for locations on the upper door, upper drag-link, and the outer wheel are presented. PSD sensitivity to mesh refinement is shown on the left column of the Figure. The results compare quite well to the UFAFF experiment (black line), and show improvement with refinement. Low frequency differences between LAVA-LBM and the experimental PSD results may be attributed to: 1) omitting inflow background turbulence levels, 2) including no-slip wall boundary conditions and under-resolving boundary layers and separation. For sensor 4, the results compare well with the contributed CFD simulations, especially the solution from Exa.[49] Sensor 13 LAVA-LBM results appear to under-predict the UFAFF PSD levels, but the results are still comparable to the other CFD results. For frequencies in the 10kHz level, LAVA-LBM consistently gives excellent results as compared to other CFD contributors.

Figure 8 compares LAVA results based on different numerical schemes on the upper drag-link (sensor 5). Figure 8(a) compares linear bounceback (LBB) and standard bounceback (SBB) with UFAFF observations. It is clear that SBB has a dramatic loss in PSD across all frequencies. This is likely due to SBB damping pressure oscillations on the surface.

Figure 8(b) compares the results of two different LAVA Cartesian Navier-Stokes WENO schemes, namely WENO5 and WENO6, with LB-3. Considering that the surface Δx of LB-3 is 4 times coarser than that of NS-3-GCM, the PSD accuracy is striking. We expect that a local mesh refinement targeted at critical flow features, combined with recursively sub-cycled SAMR (SAMR is already in the code, but not used for this paper), will both improve efficiency and increase accuracy.

III.C. Flow Structures

Isosurfaces of vorticity in the wake of the landing gear are visualized in Figure 9. As expected, the finer meshes can support progressively finer flow features. Run LB-1 is far too coarse to resolve the vortex cores at the prescribed contour levels, while LB-2 begins to capture the overall large scale flow features including the wheel wake, door blockage, and flow separation from the shock strut. LB-3 provides a visible improvement in both the large and small scale structures. LB-4 appears to only improve the prediction of finest scales



(a) LB-1 (b) LB-2

(c) LB-3 and LB-3-IO (d) LB-4

Figure 5. Computational meshes used in the simulations.

compared to LB-3 while leaving the overall details of the simulated flow largely unchanged. It is also apparent that transition to turbulence on the wheel hub occurs earlier on finer meshes.

Figure 10 compares isocontours of velocity magnitude in the center plane from the following simulations - (a) 0.24mm surface and 1mm wake mesh Navier-Stokes (NS-3-GCM), (b) 1mm surface and wake mesh Navier-Stokes (NS-3-IIM), (c) 4mm surface and wake mesh Lattice Boltzmann (LB-1), (d) 2mm surface and wake mesh Lattice Boltzmann (LB-2), (e) 1mm surface and wake mesh Lattice Boltzmann (LB-3), and (f) 0.5mm surface and wake mesh Lattice Boltzmann (LB-4). For identical near-surface and near-wake mesh resolution, the LBM simulation is able to capture dramatically finer structures than the NS simulation because of lower numerical dissipation, NS-3-IIM vs LB-3. With enhanced near-surface meshing, the flow structures of NS-3-GCM improve but are still much more dissipative than LB-3, which is 12x faster. Figure 10 shows the incredibly rich flow structure captured by the LB-4 simulation. The LB-4 run shows a level of detail that is unprecedented for a landing gear noise simulation.

III.D. Performance

Performance for different cases is compared in the Table 1, including wall clock time (measured in days) and the CPU core hours (also shown in days) needed to simulate the 0.19s of physical time recommended by the workshop committee to ensure adequate convergence of the relevant statistics. Relative SBU expense is expressed in terms of core days normalized relative to LB-3-IO run time. All runs utilized Broadwell cores on NASA's Pleiades supercomputer, except where the use of Ivy Bridge or Haswell cores are noted. Relative SBU expense takes into account processor performance ratios: Broadwell vs Ivy Bridge where a factor of $4.04/2.51 = 1.61$ is applied, or Broadwell vs Haswell where a factor of $4.04/3.34 = 1.21$ is applied. These ratios are reflected in the processor standard billing units (SBU) on Pleiades. For a similar grid size and accuracy levels (see Figure 10), the speedup of LB over NS is roughly a factor of 12. The NS cell count was somewhat greater than that of LB (at about 10%) and the NS grid resolution was finer in the immediate vicinity of the geometry for the NS run, yet this speedup is significant given predictions of comparable or superior accuracy were obtained by LB.

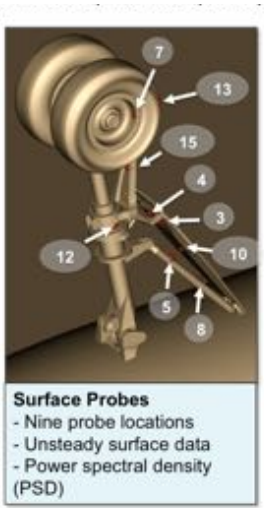


Figure 6. Data extraction locations showing point probe locations.

Comparing the run LB-4 to NS-3-GCM in Table 1, we see that the LB simulation on a 1.6 billion cell grid is roughly 50% faster (i.e. 33% less Relative SBU Expense) than the NS simulations on a 298 million cell grid on a slightly fewer number of cores. The LAVA-LBM code's ability to perform the 1.6 billion point calculation even on a relatively modest number of processors in a meaningful turnaround time is a key enabling technology that we anticipate will open the door to wall-modeled large eddy simulations of unprecedented resolution for practical engineering configurations at full-scale Reynolds numbers in the near future.

A comparison of runs LB-3 and LB-3-IO in Table 1 shows that Relative SBU Expense increases by 50% when surface probes are sampled and written to disk every 10 steps instead of every 1000, which suggests that the surface probe routines require further attention. In particular, whereas sampling to the surfaces is a parallel operation, the output to disk is currently a serial operation. This limitation will be addressed in the near future, such as in the manner of the already parallelized volume output, which leverages structured Cartesian domain partitioning and the well established parallel HDF5 library.[50]

IV. Conclusion

In this paper, LBM and Cartesian NS approaches within the Launch Ascent and Vehicle Aerodynamics (LAVA) software infrastructure were applied to simulate flow past a landing gear. Surface pressure fluctuations from the simulations were compared to those measured using flush-mounted transducers at several strategic locations on the primary components of the gear where the primary noise sources are believed to be located. The pressure fluctuations predicted by the two approaches are found to be largely consistent. The numerical experiments reported here indicate that the LB approach achieves an approximately fifteen fold speed-up over the higher-order NS discretization without any penalty in accuracy. The LB approach is vectorizable, with significantly fewer floating point operations than WENO+RK4 and greatly improved memory traffic.[21] In general, the LB approach minimizes numerical dissipation while simultaneously retaining the robustness necessary for under-resolved engineering simulations at full-scale Reynolds numbers.

Our future efforts will focus on improved wall modeling for arbitrarily complex geometry at high Reynolds numbers. Initial progress towards a moving geometry capability within LBM solver in LAVA is also underway, and takes advantage of the progress made within the LAVA Cartesian Navier-Stokes codebase.[45] Extension of the LBM solver for high speed flows is in progress. A significant effort to optimize the performance of the LAVA-LBM codebase is planned in the near future. Reducing time-to-solution for a desired level of accuracy within the context of airframe noise prediction problems is a key enabling technology for several mission critical applications that currently rely on semi-empirical approaches.

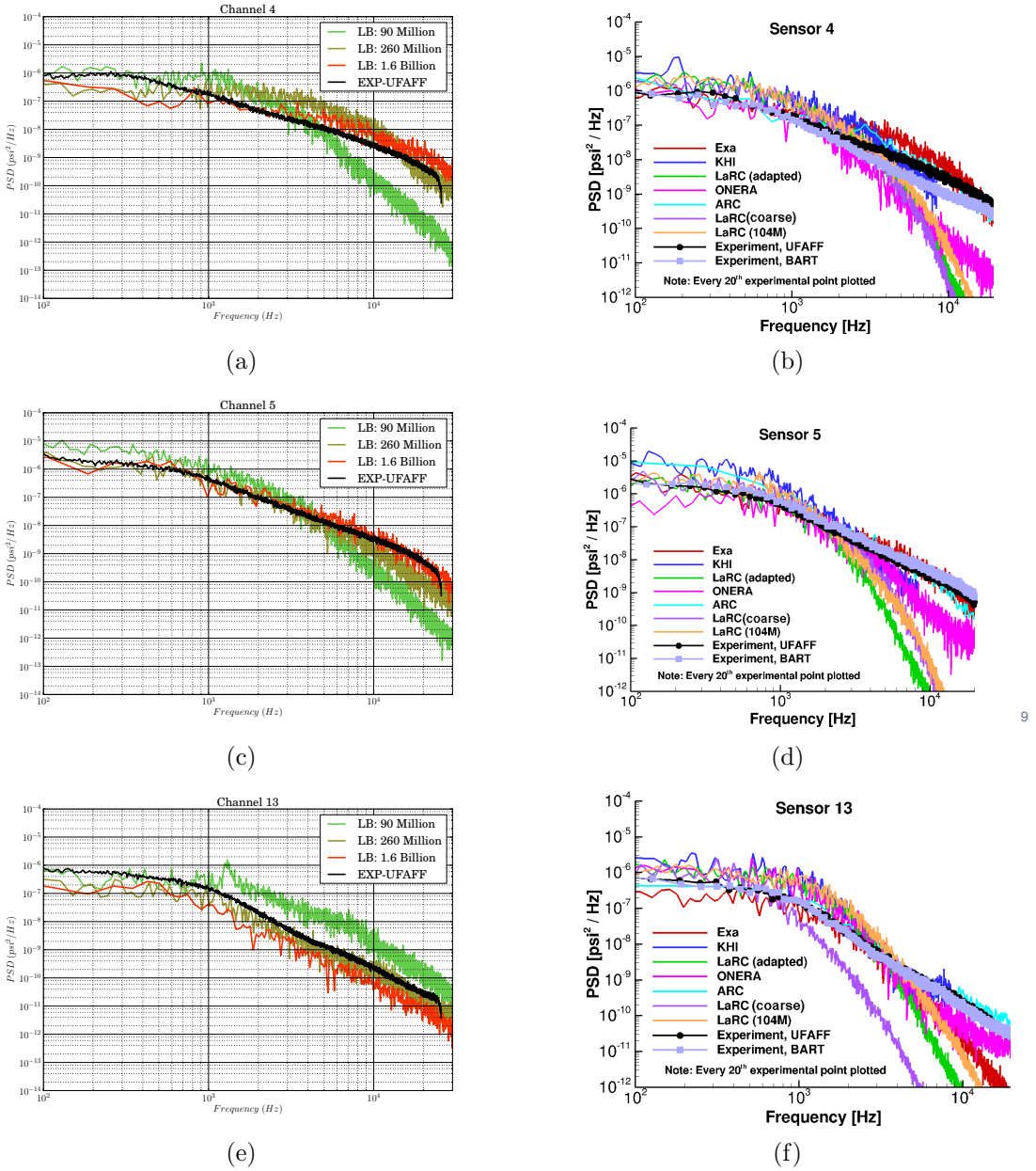
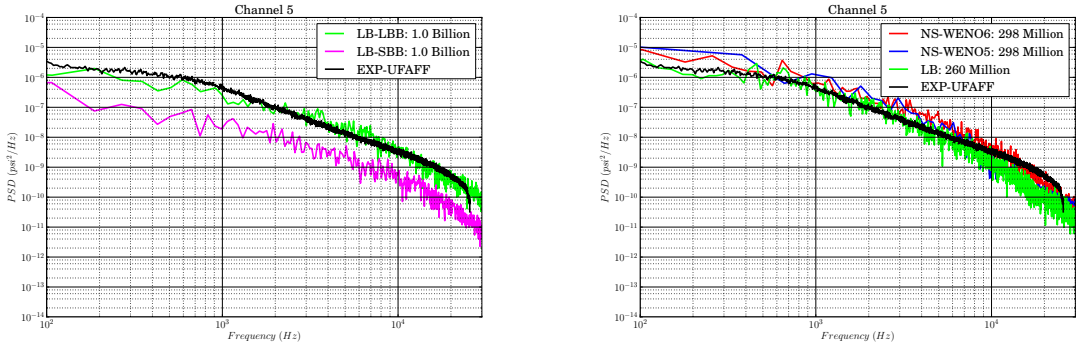


Figure 7. Power spectral density of sound at different locations: (a) and (b) probe 4 on the upper door, (c) and (d) probe 5 on the upper drag-link, (e) and (f) probe 13 on the outer wheel. The left column shows grid refinement study using LAVA LBM, while the right column shows data taken from a presentation by the BANCIII workshop organizer.[49]

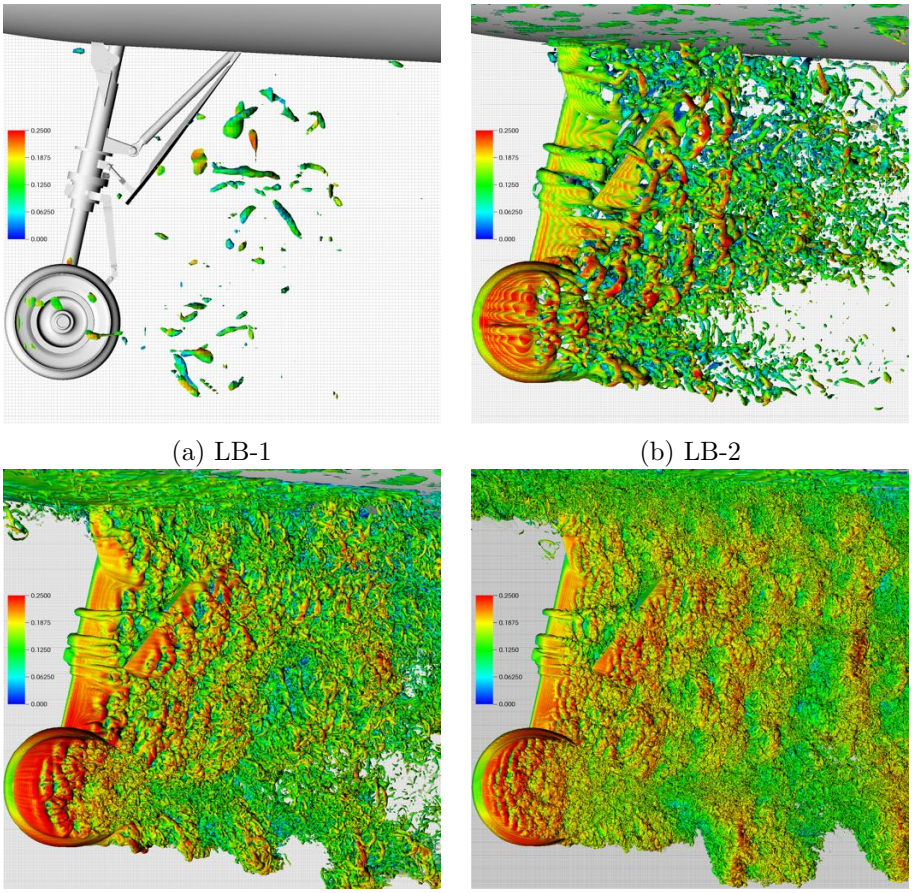


(a) (b)

Figure 8. Power spectral density of sound at probe 5 on the upper drag link. (a) shows sensitivity to LBM boundary condition (SBB vs LBB) LAVA LBM, while (b) shows a comparison with Navier-Stokes solutions using WENO5 and WENO6.

V. Acknowledgments

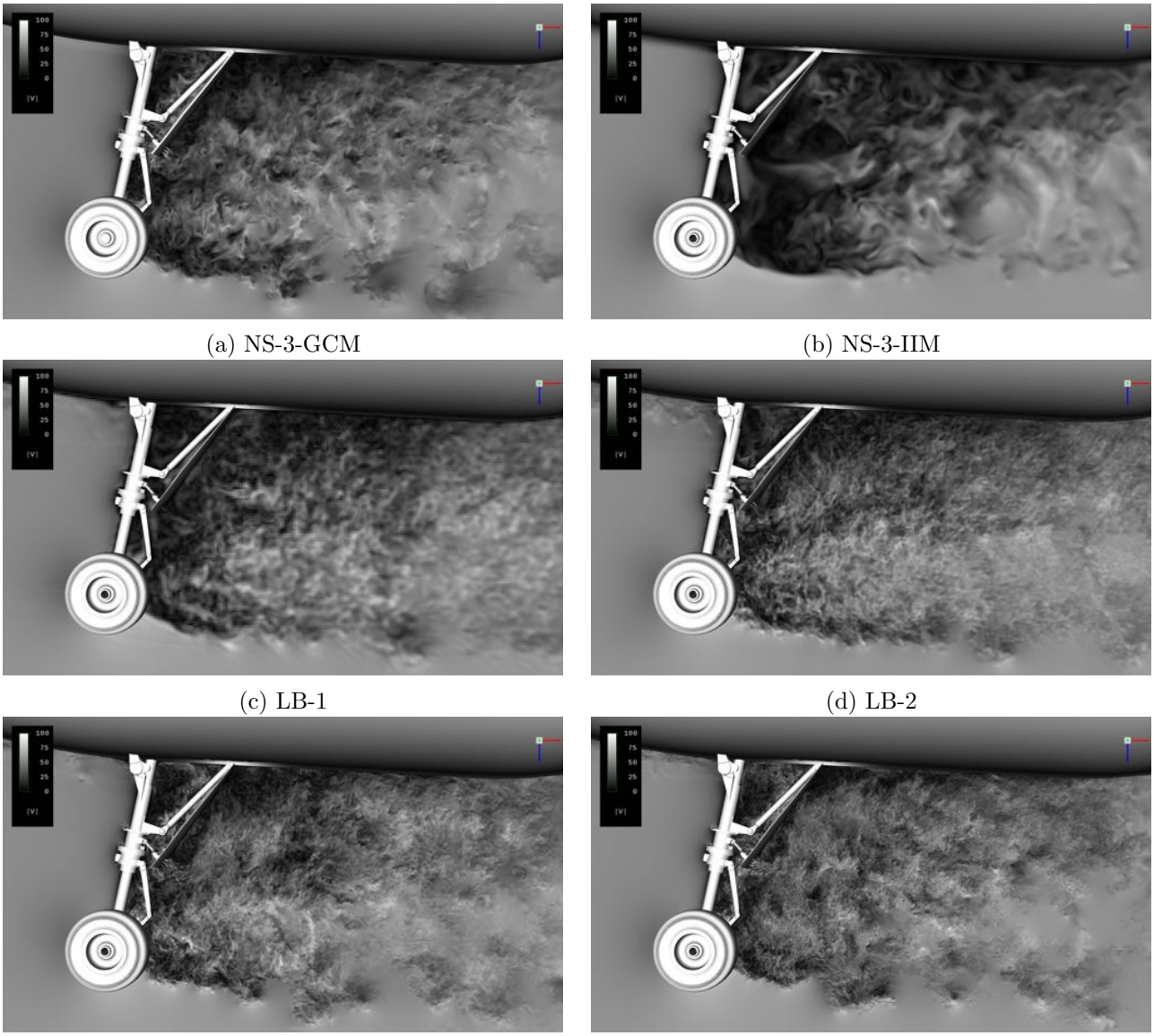
This work is partially supported by NASA's Transformational Tools and Technologies (TTT) and Advanced Air Transport Technology (AATT) projects. The authors would like to thank Timothy Sandstrom of NASA Ames Research Center for the Embree interface and visualizations. Computer time has been provided by the NASA Advanced Supercomputing (NAS) facility at NASA Ames Research Center.



(a) LB-1 (b) LB-2

(c) LB-3 (d) LB-4

Figure 9. Vorticity contours on four different mesh levels showing mesh sensitivity. Colors indicate Mach number.



(a) NS-3-GCM (b) NS-3-IIM

(c) LB-1 (d) LB-2

(e) LB-3 (f) LB-4

Figure 10. Centerline slices showing velocity magnitude: (a) 0.24mm surface and 1mm wake mesh Navier-Stokes (NS-3-GCM), (b) 1mm surface and wake mesh Navier-Stokes (NS-3-IIM), (c) 4mm surface and wake mesh Lattice Boltzmann (LB-1), (d) 2mm surface and wake mesh Lattice Boltzmann (LB-2), (e) 1mm surface and wake mesh Lattice Boltzmann (LB-3), and (f) 0.5mm surface and wake mesh Lattice Boltzmann (LB-4).

References

[1]Lockhard, D. P. and Lilley, G. M., "The airframe noise reduction challenge," 2004.

[2]Farassat, F. and Casper, J. H., "Towards an airframe noise prediction methodology: Survey of current approaches," *AIAA Paper*, Vol. 210, 2006, pp. 2006.

[3]Dobrzynski, W., Ewert, R., Pott-Pollenske, M., Herr, M., and Delfs, J., "Research at DLR towards airframe noise prediction and reduction," *Aerospace Science and Technology*, Vol. 12, No. 1, 2008, pp. 80–90.

[4]Berger, M. and Oliger, J., "Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations," *J. Comput. Phys.*, Vol. 53, March 1984, pp. 484–512.

[5]Berger, M. J. and Colella, P., "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *J. Comput. Phys.*, Vol. 82, No. 1, May 1989, pp. 64–84.

[6]Almgren, A. S., Bell, J. B., Colella, P., Howell, L. H., and Welcome, M. L., "A Conservative Adaptive Projection Method for the Variable Density Incompressible Navier-Stokes Equations," *J. Comp. Phys.*, Vol. 142, 1998, pp. 1–46.

[7]Barad, M. F. and Colella, P., "A Fourth-Order Accurate Local Refinement Method for Poisson's Equation," *J. Comp. Phys.*, Vol. 209, No. 1, October 2005, pp. 1–18.

[8]Barad, M. F., Colella, P., and Schladow, S. G., "An Adaptive Cut-Cell Method for Environmental Fluid Mechanics," *Int. J. Numer. Meth. Fluids*, Vol. 60, No. 5, 2009, pp. 473–514.

[9]Zhang, Q., Johansen, H., and Colella, P., "A Fourth-Order Accurate Finite-Volume Method with Structured Adaptive Mesh Refinement for Solving the Advection-Diffusion Equation," *SIAM Journal on Scientific Computing*, Vol. 34, No. 2, 2012, pp. 179–201.

[10]Aftosmis, M. J., Berger, M. J., and Saltzman, J. S., "Robust and efficient Cartesian mesh generation for component-base geometry," *AIAA Journal*, Vol. 36, No. 6, June 1998, pp. 952–960.

[11]Nemec, M. and Aftosmis, M., "Adjoint-Based Adaptive Mesh Refinement for Complex Geometries," *46th AIAA Aerospace Sciences Meeting, Reno, NV*, Jan 7-10 2008.

[12]Mavriplis, D., "Three-dimensional unstructured multigrid for the Euler equations," *AIAA journal*, Vol. 30, No. 7, 1992, pp. 1753–1761.

[13]Barth, T. J., "Aspects of unstructured grids and finite-volume solvers for the Euler and Navier-Stokes equations," *In AGARD, Special Course on Unstructured Grid Methods for Advection Dominated Flows 61 p (SEE N92-27671 18-34)*, Vol. 1, 1992.

[14]Rogers, S. E., Kwak, D., and Kiris, C., "Steady and unsteady solutions of the incompressible Navier-Stokes equations," *AIAA Journal*, Vol. 29, April 1991, pp. 603–610.

[15]Nichols, R. and Buning, P., "User's Manual for OVERFLOW 2.1," Version 2.1t.

[16]Kiris, C. C., Housman, J. A., Barad, M. F., Brehm, C., Sozer, E., and Moini-Yekta, S., "Computational framework for Launch, Ascent, and Vehicle Aerodynamics (LAVA)," *Aerospace Science and Technology*, Vol. 55, 2016, pp. 189 – 219.

[17]Dubey, A., Almgren, A., Bell, J., Berzins, M., Brandt, S., Bryan, G., Colella, P., Graves, D., Lijewski, M., Löffler, F., et al., "A survey of high level frameworks in block-structured adaptive mesh refinement packages," *Journal of Parallel and Distributed Computing*, Vol. 74, No. 12, 2014, pp. 3217–3227.

[18]Chen, S. and Doolen, G. D., "Lattice Boltzmann method for fluid flows," *Annual review of fluid mechanics*, Vol. 30, No. 1, 1998, pp. 329–364.

[19]Yu, D., Mei, R., Luo, L.-S., and Shyy, W., "Viscous flow computations with the method of lattice Boltzmann equation," *Progress in Aerospace Sciences*, Vol. 39, No. 5, 2003, pp. 329–367.

[20]Aidun, C. K. and Clausen, J. R., "Lattice-Boltzmann method for complex flows," *Annual review of fluid mechanics*, Vol. 42, 2010, pp. 439–472.

[21]Kocheemoolayil, J., Barad, M., and Kiris, C., "How good is the Lattice Boltzmann Method?" *69th Annual Meeting of the APS Division of Fluid Dynamics*, 2016.

[22]Schornbaum, F. and Rude, U., "Massively Parallel Algorithms for the Lattice Boltzmann Method on NonUniform Grids," *SIAM Journal on Scientific Computing*, Vol. 38, No. 2, 2016, pp. C96–C126.

[23]Mittal, R., Dong, H., Bozkurttas, M., Najjar, F., Vargas, A., and von Loebbecke, A., "A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries," *Journal of Computational Physics*, Vol. 227, No. 10, 2008, pp. 4825 – 4852.

[24]Nakahashi, K., "Immersed Boundary Method for Compressible Euler Equations in the Building-Cube Method," *29th AIAA Applied Aerodynamics Conference*, June 2011, AIAA-2011-3386.

[25]Brehm, C. and Fasel, H., "A novel concept for the design of immersed interface methods," *J. Comput. Phys.*, Vol. 242, No. 0, 2013, pp. 234 – 267.

[26]Brehm, C., Hader, C., and Fasel, H., "A Locally Stabilized Immersed Boundary Method for the Compressible Navier-Stokes Equations," *J. Comput. Phys.*, Vol. 295, 2015, pp. 475 – 504.

[27]Ruffin, S. M. and Lee, J., "Adaptation of a k-epsilon Model to a Cartesian Grid Based Methodology," *International Journal of Mathematical Models and Methods in Applied Sciences*, Vol. 3, 2009, pp. 238–245.

[28]Kawai, S. and Larsson, J., "Wall-modeling in large eddy simulation: Length scales, grid resolution, and accuracy," *Physics of Fluids*, Vol. 24, No. 1, 2012, pp. 015105.

[29]Le Bras, S., Deniau, H., Bogey, C., and Daviller, G., "Development of compressible large-eddy simulations combining high-order schemes and wall modeling," *21st AIAA/CEAS Aeroacoustics Conference*, 2015, p. 3135.

[30]Aikens, K. M., Dhamankar, N. S., Martha, C. S., Situ, Y., Blaisdell, G. A., Lyrintzis, A. S., and Li, Z., "Equilibrium wall model for large eddy simulations of jets for aeroacoustics," *52nd Aerospace Sciences Meeting*, 2014, p. 0180.

[31]Reichardt, H., "The fundamentals of turbulent heat transfer," *Archiv fuer die Gesamte Waermetechnik,(West Germany)*, Vol. 6, 1951, pp. 129–143.

[32]Chapman, S. and Cowling, T. G., *The mathematical theory of non-uniform gases: an account of the kinetic theory of viscosity, thermal conduction and diffusion in gases*, Cambridge university press, 1970.

[33]Bhatnagar, P. L., Gross, E. P., and Krook, M., "A Model for Collision Processes in Gases. I. Small Amplitude Processes in Charged and Neutral One-Component Systems," *Physical Review E*, Vol. 94, 1954, pp. 511–525.

[34]Li, Y., Shock, R., Zhang, R., and Chen, H., "Numerical study of flow past an impulsively started cylinder by the lattice-Boltzmann method," *Journal of Fluid Mechanics*, Vol. 519, 2004, pp. 273–300.

[35]S. Ansumali, I. V. K. and ttinger, H. C., "Minimal entropic kinetic models for hydrodynamics," *Europhysics Letters*, Vol. 63, No. 6, 2003, pp. 798–804.

[36]Lallemand, P. and Luo, L.-S., "Theory of the lattice Boltzmann method: Dispersion, dissipation, isotropy, Galilean invariance, and stability," *Physical Review E*, Vol. 61, Jun 2000, pp. 6546–6562.

[37]Bosch, F., Chikatamarla, S. S., and Karlin, I. V., "Entropic multirelaxation lattice Boltzmann models for turbulent flows," *Physical Review E*, Vol. 92, 2015, pp. 043309.

[38]Latt, J. and Chopard, B., "Lattice Boltzmann method with regularized pre-collision distribution functions," *Mathematics and Computers in Simulation*, Vol. 72, No. 26, 2006, pp. 165 – 168.

[39]Smagorisnky, J., "General circulation experiments with the primitive equations," *Monthly Weather Review*, Vol. 91, No. 3, 1963, pp. 99–164.

[40]Mhamed Bouzidi, M. F. and Lallemand, P., "Momentum transfer of a Boltzmann-lattice fluid with boundaries," *Physics of Fluids*, Vol. 13, No. 11, 2001, pp. 3452–3459.

[41]Berger, M. J. and Oliger, J. E., "Adaptive mesh refinement for hyperbolic partial differential equations," Tech. rep., Stanford University, Stanford, CA, USA, 1983.

[42]Colella, P., Graves, D. T., Ligocki, T. J., Martin, D. F., Modiano, D., Serafini, D. B., and Straalen, B. V., "Chombo Software Package for AMR Applications - Design Document," unpublished.

[43]Martin, D. F., Colella, P., and Graves, D., "A Cell-Centered Adaptive Projection Method for the Incompressible Navier-Stokes Equations in Three Dimensions," *J. Comput. Phys.*, Vol. 227, 2008, pp. 1863–1886.

[44]Ligocki, T. J., Schwartz, P. O., Percelay, J., and Colella, P., "Embedded boundary grid generation using the divergence theorem, implicit functions, and constructive solid geometry," *Journal of Physics: Conference Series*, Vol. 125, Institute of Physics Publishing, 2008, p. 012080.

[45]Brehm, C., Barad, M. F., and Kiris, C. C., "Open Rotor Computational Aeroacoustic Analysis with an Immersed Boundary Method," *54th AIAA Aerospace Sciences Meeting*, 2016, p. 0815.

[46]Sethian, J. A., *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, Vol. 3, Cambridge university press, 1999.

[47]Zhao, H., "A fast sweeping method for eikonal equations," *Mathematics of computation*, Vol. 74, No. 250, 2005, pp. 603–627.

[48]Wald, I., Woop, S., Benthin, C., Johnson, G. S., and Ernst, M., "Embree: A kernel framework for efficient cpu ray tracing," *ACM Transactions on Graphics (TOG)*, Vol. 33, No. 4, 2014, pp. 143.

[49]Khorrami, M. R., "An Overview of Contributions for Nose Landing Gear Configuration-BANC-III Workshop," *21st AIAA/CEAS Aeroacoustics Conference 22-26 June 2015, Dallas, Texas*, 2015.

[50]Folk, M., Heber, G., Koziol, Q., Pourmal, E., and Robinson, D., "An overview of the HDF5 technology suite and its applications," *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, ACM, 2011, pp. 36–47.